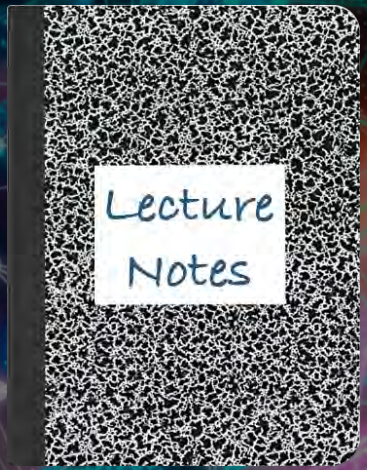
CS 419: Computer Security

Week 10: Network Security

Part 2: Network & Transport Layers



Paul Krzyzanowski

© 2026 Paul Krzyzanowski. No part of this content may be reproduced or reposted in whole or in part in any manner without the permission of the copyright owner.

Routing Between Networks

The network layer, IP, and BGP

On-Path And Off-Path Attackers

- An **on-path** attacker is somewhere the victim's packets pass through
  - A compromised router, or the gateway position from the local network attacks
  - Can see and alter the packets
- An **off-path** attacker is anywhere else on the Internet
  - Can send packets but cannot see the victim's
  - Has to guess anything it needs to know

Every attack we cover will be one or the other

The Network Layer: IP

- Responsible for delivering packets from one host to another across networks
  - No guarantees on ordering or delivery
- **Routing**: each host knows one or more routers; routers know how to reach other networks
- **Fragmentation**: a packet is split if a link's maximum size (MTU) is too small, and reassembled at the destination
- **Error reporting**: ICMP messages back to the sender
- **Time-to-live** (TTL): a hop count against loops

*MTU = Maximum Transmission Unit = maximum packet size on a network link

*ICMP = Internet Control Message Protocol = An IP protocol used by devices to send error and diagnostic info

Forging Source Addresses

Goal:
Hide, impersonate a trusted host, or aim a server's replies at a victim

- Routers forward by destination address and most do not check the source
- The limit: **replies go to the forged address, not the attacker**
  - A spoofed packet cannot complete a login, but it can make a server reply to someone else
- **Recommended fix: Ingress filtering** (BCP 38)
  - Drop packets leaving a network whose source address does not belong to it
  - The fix is voluntary and uneven.
    - An attacker needs only one network that does not filter

Routers As Attack Targets

Goal: an on-path position for everything that passes through the router

Routers are just special-purpose computers

- Insecure configurations
- Router OS vulnerabilities
- Extra services

Subject to attacks:

- **Denial of Service (DOS)**
  - Flood the router (e.g., lots of ICMP packets from lots of sources)
- **Routing table poisoning**
  - Either by breaking into a router or by sending modified routing data update packets
- **Malware**
  - Either by breaking into a router or by sending modified routing data update packets

A compromised router sees every packet and chooses where they go

Thousands of Asus routers are being hit with stealthy, persistent backdoors

Backdoor giving full administrative control can survive reboots and firmware updates.

Dan Goodin • May 28, 2025

~9,000 routers affected

Thousands of home and small office routers manufactured by Asus are being infected with a stealthy backdoor that can survive reboots and firmware updates in an attack by a nation-state or another well-resourced threat actor, researchers said.

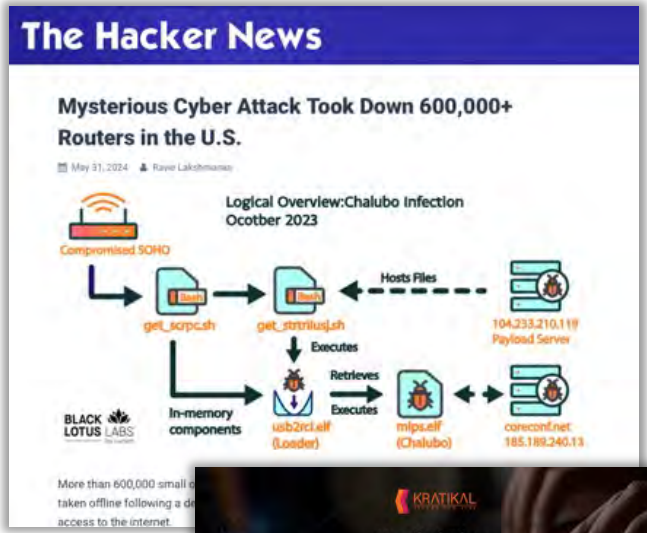
The unknown attackers gain access to the devices by exploiting now-patched vulnerabilities, some of which have never been tracked through the internationally recognized CVE system. After gaining unauthorized administrative control of the devices, the threat actor installs a public encryption key for access to the device through SSH. From then on, anyone with the private key can automatically log in to the device with administrative system rights.

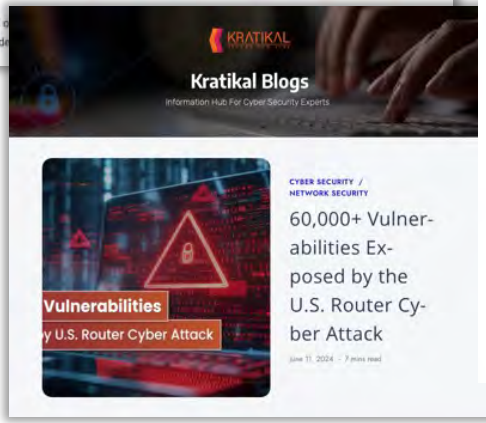
Durable control

"The attacker's access survives both reboots and firmware updates, giving them durable control over affected devices," researchers from security firm GreyNoise reported Wednesday. "The attacker maintains long-term access without dropping malware or leaving obvious traces by chaining authentication bypasses, exploiting a known vulnerability, and abusing legitimate configuration features."

https://arstechnica.com/security/2025/05/thousands-of-asus-routers-are-being-hit-with-stealthy-persistent-backdoors/

Routers are vulnerable, like computers













They often use default login credentials

	Router Brand	Default IP Address	Default Username	Default Password
1	3Com	http://192.168.1.1	admin	Admin
2	Belkin	http://192.168.2.1	admin	admin
3	BenQ	http://192.168.1.1	admin	Admin
4	D-Link	http://192.168.0.1	admin	Admin
5	Digicom	http://192.168.1.254	admin	Michelangelo
6	Linksys	http://192.168.1.1	admin	Admin
7	Netgear	http://192.168.0.1	admin	password
8	Sitecom	http://192.168.0.1	sitecom	Admin
9	Asus	http://192.168.1.1	admin	admin
10	Synology	http://192.168.1.1	admin	Admin
11	Arris	http://192.168.0.1	admin	password
12	Apple iphoneIOS4.X	http://10.0.1.1	root	alpine
13	DELL	http://192.168.1.1	admin	password
14	Huawei ADSL2+	http://192.168.0.1	admin	admin
15	Netcomm	http://192.168.1.1	admin	password

https://www.softwaretestinghelp.com/default-router-username-and-password-list/

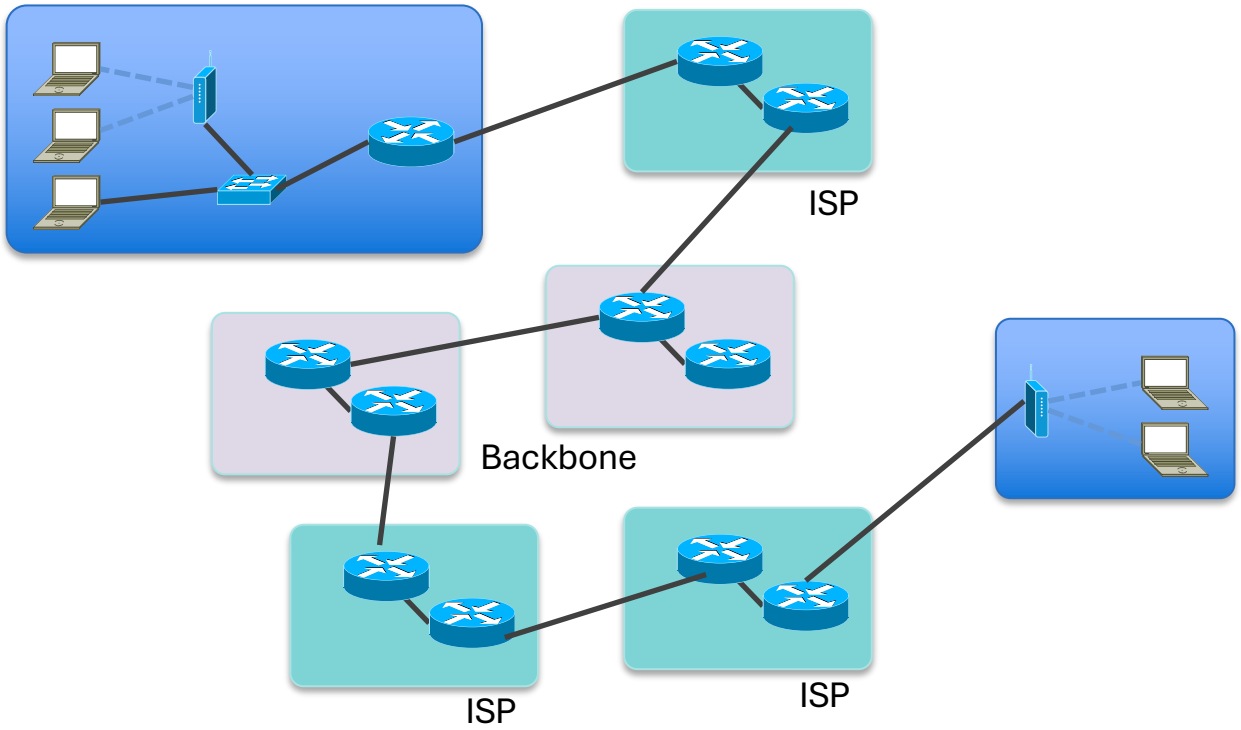
Routing With BGP

How the Internet's networks exchange routes

A Network Of Networks

Packet switching: store-and-forward routing across multiple physical networks

… across multiple organizations



BGP: The Border Gateway Protocol

- **Autonomous system** (AS)
  - Set of address prefixes and routers under one administration
  - An ISP, university, cloud provider, large company (Rutgers is AS 46)
  - More than 75,000 of them
- Autonomous systems exchange routes with **BGP**
  - An AS announces each prefix it owns to its neighbors
  - A neighbor prepends its own AS number and announces it to its neighbors
  - A remote router sees the list of ASes the route passes through: the **AS path**
- Each AS decides by policy which routes to accept and which to pass on
- Taking part requires a BGP session with a neighboring AS.
  An ordinary host cannot announce a route

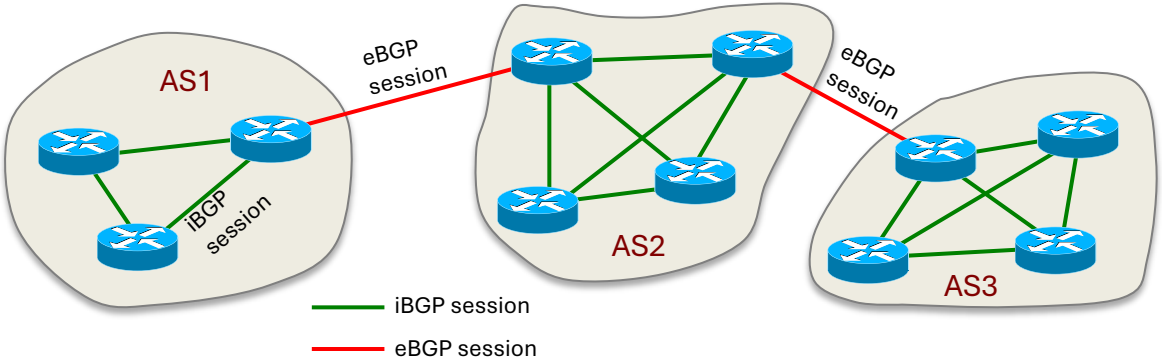
A **prefix** is a range of addresses with the the same leading bits.

Rutgers has 128.6.0.0/16

The /16 means tells a router that any address where the top (high-order) 16 bits match 128.6 belongs to Rutgers.

It's more efficient than listing out 65,536 IP addresses.

BGP Sessions Run Over TCP

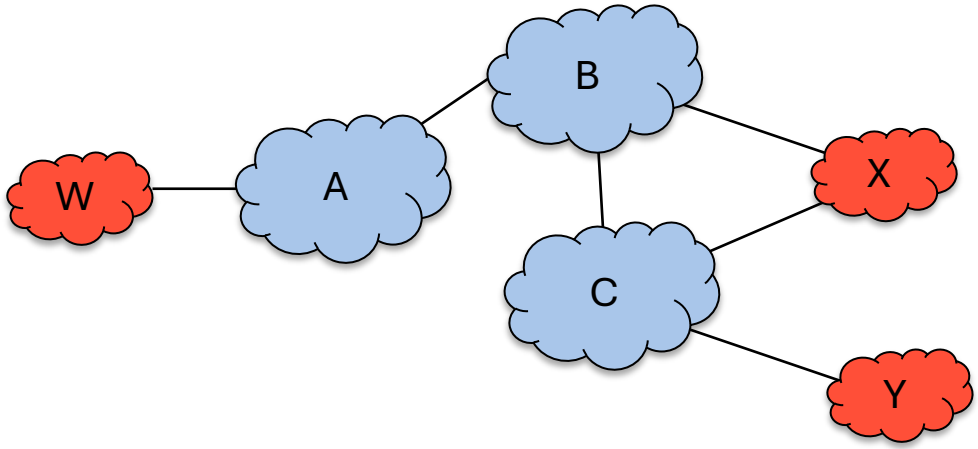


Pairs of routers exchange information via semi-permanent TCP connections

- One connection for each link between gateway routers
  - External BGP (eBGP) session
- Also, BGP TCP connections between routers *inside* an AS
  - Internal BGP (iBGP) session

Transit And Stub Networks

- A, B, C: transit ASes – ISPs & backbone
- W, X, Y: stub ASes – customers



The Most Specific Prefix Wins

- A router picks the **most specific prefix** that matches the destination
  - `192.0.2.0/25` (128 addresses) wins over `192.0.2.0/24` (256 addresses)
  - A more specific address wins, regardless of path length or who announced it
- Policy and path length conisdered only among routes to the same prefix
- **Every BGP hijack we cover exploits this rule**

BGP Hijacking

Goal: pull traffic for someone else's addresses into the attacker's network

- **Requires**: routing access
  (a provider that does not filter, a compromised router, or an AS)
- BGP does not verify that an AS may announce a prefix, or that the AS path is real.
  - An unfiltered announcement spreads worldwide in seconds
- A **BGP hijack** is an unauthorized announcement of someone else's addresses
  - A **more specific prefix**: the victim announces a /22, the attacker a /24
  - Every router will prefer the attacker's prefix

What The Hijacker Does With The Traffic

Once the traffic arrives, the hijacker can do three things with it:

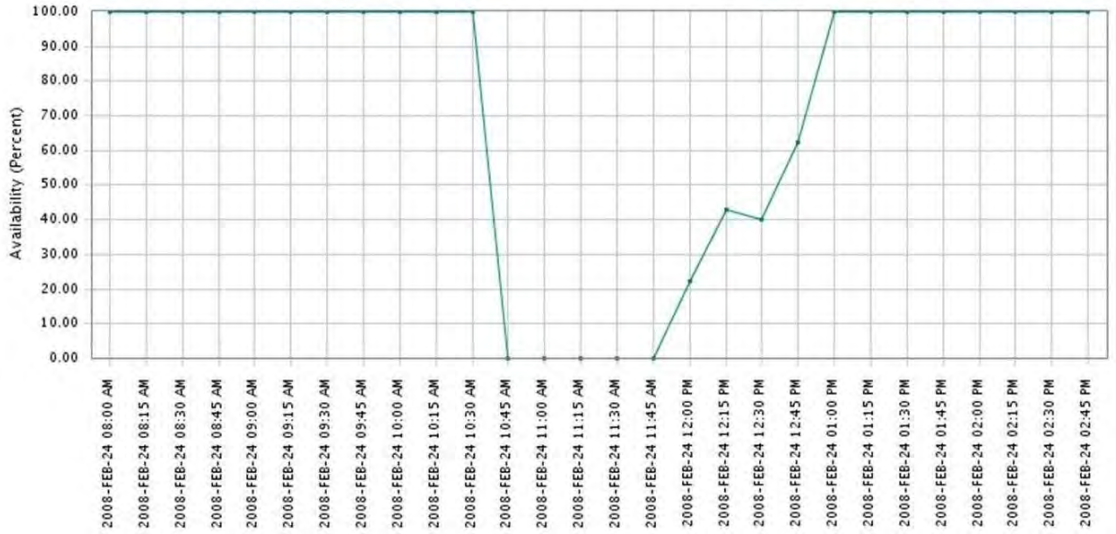
1. **Drop it**. The addresses become unreachable: a denial of service
2. **Inspect it and forward it** to the real destination: an adversary-in-the-middle position
3. **Respond to it**. The hijacker impersonates the real service

Related failure:

- A **route leak**: an AS passes on routes it learned
- Traffic flows through a network that was never meant to transport it

YouTube, February 2008

- YouTube was unreachable for most of the world for about two hours
- Pakistan Telecom, ordered to block YouTube inside Pakistan, announced a route for YouTube's addresses. Its international provider passed it to the world

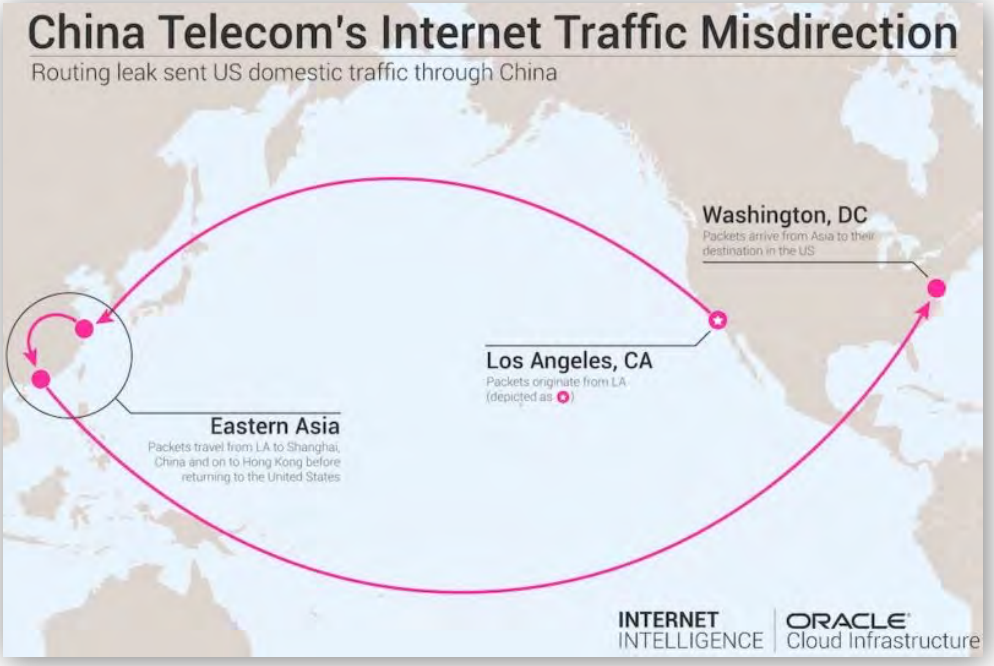


YouTube, February 2008

- Pakistan Telecom announced `208.65.153.0/24` (256 addresses)
- YouTube announced `208.65.152.0/22` (1,024 addresses)
  - The /24 was more specific, so every router that accepted it preferred it
  - Within minutes, YouTube's traffic flowed to Pakistan and was dropped
- YouTube responded with its own /24, then with /25 prefixes
  - PCCW withdrew the bad route about two hours after it appeared
- *A configuration decision, not an attack*

China Telecom, 2015 To 2018

- China Telecom redirected large chunks of Internet traffic through their routers
- This took place for 2.5 years
- China Telecom was incorrectly advertising routes to Verizon's Asia-Pacific AS (AS703)
  - Packets would be routed to China
- Could have been a configuration error



https://arstechnica.com/information-technology/2018/11/strange-snafu-misroutes-domestic-us-internet-traffic-through-china-telecom/

MyEtherWallet, April 2018

- An attacker announced five /24 prefixes belonging to Amazon's Route 53 DNS service
  - Sent from an Ohio provider's AS
  - Large transit providers accepted them
  - **For about two hours, queries for Amazon's name servers went to the attacker**
- The attacker responded to one query
  - `myetherwallet.com`, a site for managing Ethereum wallets
  - The response pointed to a copy of the site hosted in Russia
- Some users clicked through the browser's invalid certificate warning
  - About $150,000 stolen
- **Certificate validation was the only defense (a warning) that held**

Softaculous, August 2026

- Hetzner, a large German hosting provider, announces 162.55.0.0/16.
  - A small hosting network announced 162.55.80.0/24, a smaller block inside it
  - The addresses belonged to Softaculous, which sells server management software to web hosts
- Active for about 22 hours within a 33-hour window
- The attacker obtained a **valid certificate**
  - Browsers and update clients saw a valid certificate and didn't raise a warning
- Servers that checked for updates received a backdoored package.

The Softaculous Chain Of Failures

1. Route hijack
2. Fraudulent domain-control check
3. Valid certificate
4. Malicious update accepted without independent verification

- Compared with 2018 attack:
  - Forged DNS wasn't needed (the hijacked addresses were the servers themselves),
  - There wasn't a certificate warning
- The company's response: **sign the update packages**
  - Stops the backdoor regardless of routing, if the verification key is already on the client and not downloaded through the hijacked path

BGP Hijacks Are Declining, But Continue

UNIQUE ROUTE LEAKER ASes	Q3, 2026	UNIQUE BGP HIJACKER ASes
1,935	JULY	4,702
1,871	AUGUST	4,415
1,773	SEPTEMBER	5,076

https://qrator.net/blog/details/Q3-2026-DDoS-bad-bots-and-BGP-incidents-statistics-and-overview/

Defending Routing: Filtering And Monitoring

- BGP cannot verify an announcement
  - Defenses add verification from outside
- **Prefix filters** on each connection to a neighbor
  - A customer's filter lists the prefixes it may announce
    - The router rejects anything else
  - Stops most accidental hijacks
  - A limit on the number of prefixes accepted from a neighbor stops a route leak
- **Monitoring** services
  - Watch announcements and alert a prefix owner when someone else announces its addresses

RPKI: Signed Route Origins

RPKI: Resource Public Key Infrastructure

Regional Internet Registries act as certificate authorities for the holders

The holder signs a **Route Origin Authorization** (ROA):
this AS may announce this prefix, up to this maximum length

- A ROA for `208.65.152.0/22` authorizes that one announcement from one AS
- Any /24 inside it is unauthorized

- Operators download the ROAs and check announcements against them

See: https://labs.ripe.net/author/niklas-vogel/crashing-the-party-vulnerabilities-in-rpki-relying-party-software/

Route Origin Validation

- A router compares each announcement with the published ROAs:
  - Passes when a covering ROA authorizes its claimed origin AS and prefix length
  - Marked **invalid** when ROAs cover the prefix and none authorize the announcement
  - Unverified when no ROA covers the prefix
- The operator's routing policy decides which routes to accept or drop
- This stops a hijack only when all three hold:
  1. The legitimate holder has published a ROA
  2. The ROA does not permit the hijacker’s origin AS or prefix length
  3. The receiving network drops routes marked invalid
- September 2026: about 70% of prefixes are covered by a ROA; major transit providers drop invalid routes

What Origin Validation Does Not Catch

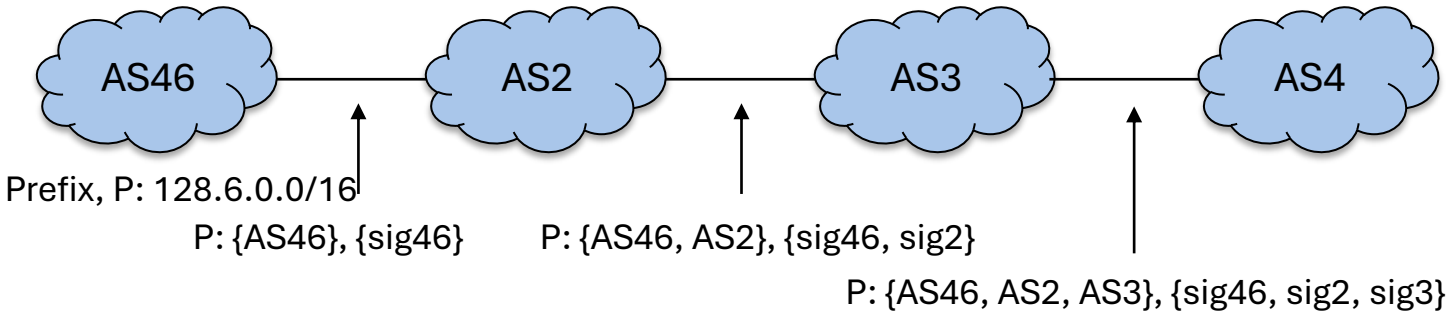
- Origin validation checks only the claimed origin AS,
  - The announcer creates this
- A forged AS path that ends in the legitimate origin AS passes
  - As long as the prefix length is also authorized
- This is what the Softaculous hijacker did
  - The forged path ended in Hetzner's AS
  - Hetzner's ROA for the /16 allowed announcements as long as /24
    - So the hijacked /24 validated
  - A ROA allowing only the /16 would have made it invalid
    - Hetzner tightened its ROAs afterward

Stopping a forged path requires verifying the path itself

Verifying The Path: BGPsec

BGPsec (2017, RFC 8205): each AS signs the announcement as it forwards it, so a receiving router can verify every hop

- Requires every AS on the path to participate. Almost entirely undeployed.



Ensures the route's path is valid as it propagates through the network

See: https://labs.ripe.net/author/niklas-vogel/crashing-the-party-vulnerabilities-in-rpki-relying-party-software/

Path Verification Is Still Rare

Origin validation checks the origin AS an announcement claims, not the path it claims to have taken

- Origin validation is widely deployed. Path verification is not
  - A forged path that ends in an authorized origin still passes almost everywhere
  - BGPsec would fix this, but it needs every AS on the path to sign

Why Routing Security Deploys Slowly

- 75,000 autonomous systems, independently operated
  - A defense helps only where both the announcing and the validating networks participate
- Publishing and maintaining authorizations takes operational effort
- A mistake in a ROA drops the operator's own legitimate routes wherever validation is enforced
  - For years, many operators judged that risk greater than the risk of being hijacked
- Coverage grew once the largest providers began rejecting invalid routes:
  - A signed prefix is then protected and an unsigned one is not

See: https://labs.ripe.net/author/niklas-vogel/crashing-the-party-vulnerabilities-in-rpki-relying-party-software/

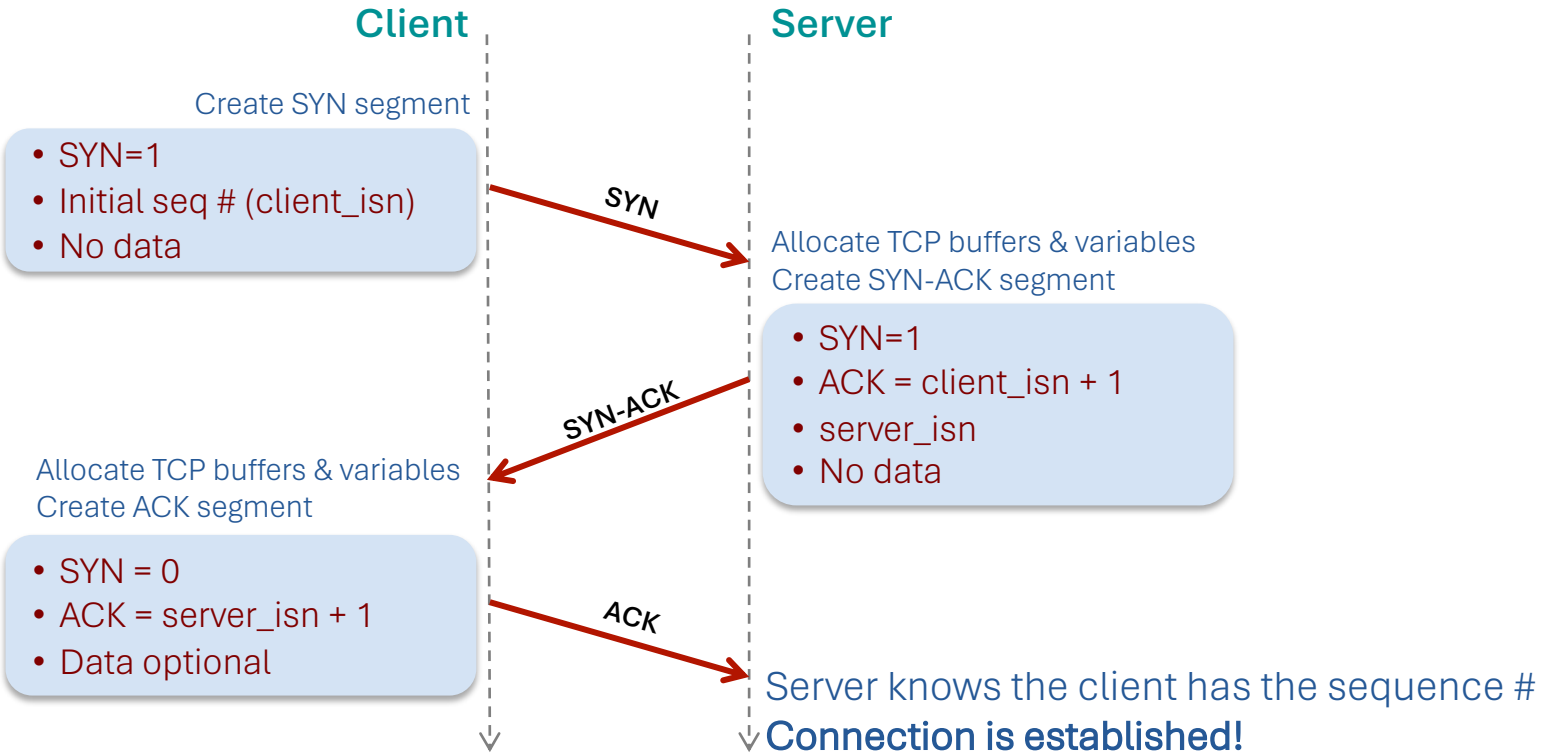
The Transport Layer

Attacks on TCP and UDP

What The Transport Layer Checks

- **Transport identifiers**:
  - Source and destination addresses and ports
  - For TCP, the sequence numbers
- An attacker who knows or can guess them can inject segments a receiver accepts
- Transport injection does not defeat cryptographic protection at the application layer
  - TLS rejects injected data: the attacker cannot produce a valid authentication tag

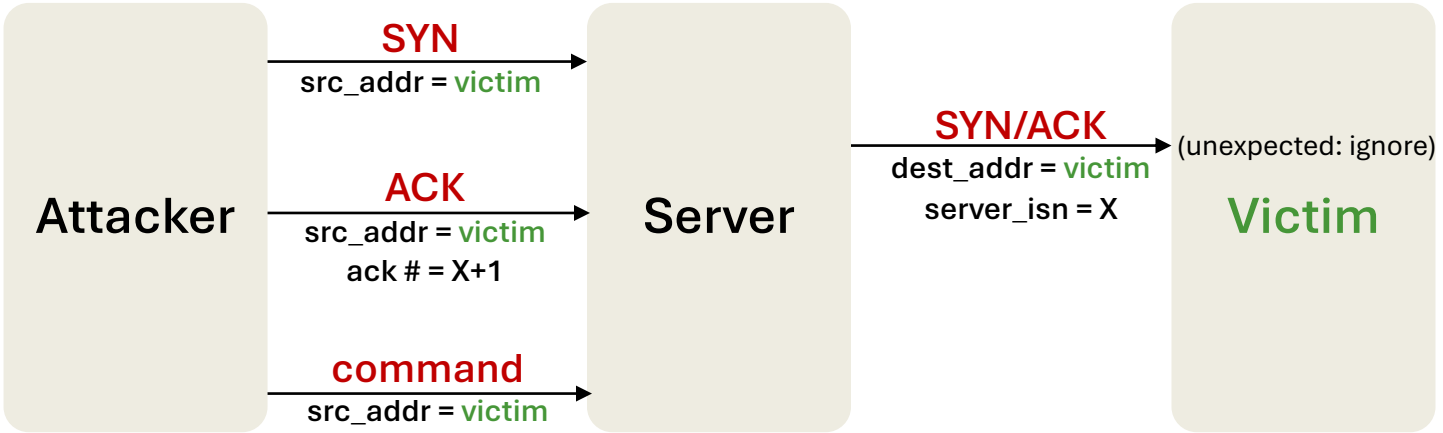
The Three-Way Handshake



Guessing Sequence Numbers

Goal: complete a handshake while claiming to be a host the server trusts

Needs: a server that trusts clients by address, and predictable initial sequence numbers



Making Sequence Numbers Unpredictable

- Compute each one from a keyed hash:
  - Hash(connection's addresses, ports, a secret)
  - Standardized as RFC 6528 (2012), from a 1996 proposal
  - An attacker cannot infer one connection's number from another's
- Blind sequence guessing is no longer practical
- An **on-path** attacker still reads the real numbers and can inject
  - Services cannot trust a client by its address
  - TLS rejects injected bytes that are not part of the encrypted stream

SYN Flooding

Goal: deny service by filling the server's table of pending connections

- On receiving a SYN packet:
  - The server allocates memory, sends a SYN-ACK, and ***waits***
  - If the ACK never arrives, the server holds the half-open connection for tens of seconds
  - The client sends just one packet
    - The server uses up memory and time
- **The attack:** send SYNs as fast as possible, usually with forged source addresses
  - The backlog fills and legitimate connection attempts are refused
- A single machine is enough for the attack
  September 1996: a SYN flood took Panix, a New York ISP, offline for days

Solution: SYN Cookies

- The server does not store the half-open connection
  - It encodes what it needs into the sequence number in the SYN-ACK
    - A value it can check later without remembering the request
    - This is called a **SYN cookie**
  - When the ACK arrives
    - The server recomputes the value from the packet's addresses and ports and compares
  - A match means the client received the SYN-ACK.
    - Only then is the connection allocated
- A SYN cookie proves the client can receive packets at its claimed address
  - It does not prove the client is honest.
    - Bots complete handshakes
  - It protects the pending-connection table, not bandwidth or the processor

Resetting Connections

Goal: terminate a connection the attacker wants stopped

- A TCP **RST flag** tells the receiver to abandon the connection immediately
  - Forge one with the right addresses, ports, and an acceptable sequence number
- An on-path attacker has everything needed
  - The Great Firewall worked this way in its early years
    - Forged resets to both ends of a connection carrying banned keywords
- An off-path attacker must know:
  the addresses and ports and guess a sequence number

Defending Against Forged Resets

- Older TCP implementations accepted a large set of sequence numbers inside the receive window
  - A few hundred thousand guesses out of four billion
- **RFC 5961** (2010):
  - A reset is accepted only if its sequence number matches the next expected one
  - A blind attacker is left with one correct value out of four billion
- Off-path resets became impractical
- The on-path attacker is untouched

Resets And Encryption

- TLS does not help here
  - An on-path attacker can drop packets or inject an accepted reset on an encrypted connection as easily as on a plain one
  - Encryption protects a connection's content, not its availability
- Connections whose loss is especially damaging, such as the BGP sessions between routers, can be protected with an option called TCP-AO
  - A message authentication code on every segment, so a forged reset is discarded
  - It authenticates the neighbor's segments, not the truth of what the neighbor announces

TCP-AO is is a security extension for TCP designed to authenticate and verify TCP segments between trusted peers. It's usually used by network operators and service providers. (You don't have to know this.)

UDP

- No handshake, sequence numbers, or connection state
  - *Nothing to check a packet against*
- A service that answers any request answers a forged one
  - A larger reply than request: **amplification** (covered with denial of service)
- A UDP application can add checks
  - An echoed token proves the client can receive packets at its address
  - Authenticating contents takes a cryptographic tag or signature
  - QUIC (used in HTTP/3) builds connections, ordering, and encryption on UDP

The Domain Name System

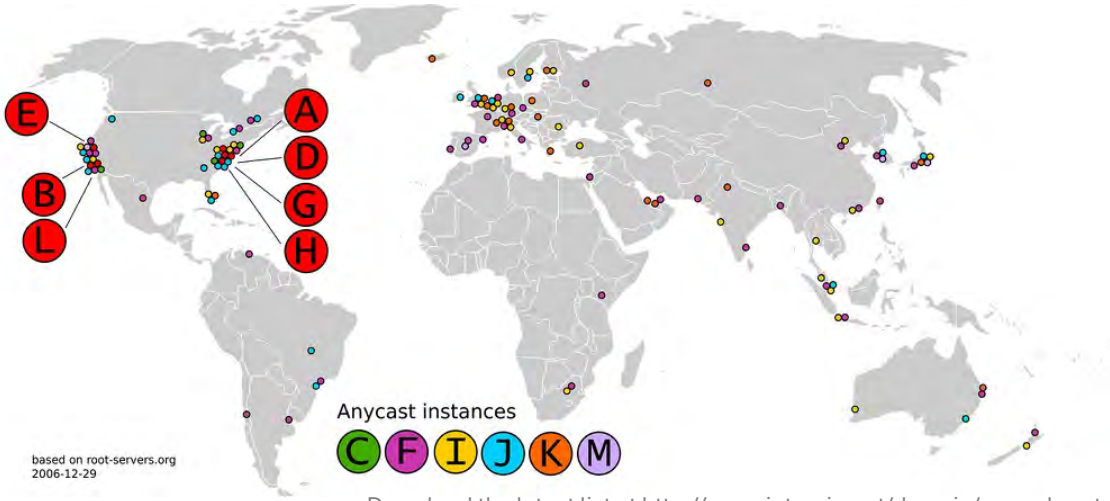
Pharming, cache poisoning, DNSSEC, rebinding, and hijacked domains

The Domain Name System

- A hierarchical service that maps domain names to IP addresses
  - `www.cs.rutgers.edu`: host `www`, in `cs.rutgers.edu`, in `rutgers.edu`, in `edu`, in the root
- The name space is divided into **zones**, each served by its own servers
  - A zone's servers answer for its names and know where to find the servers for anything **delegated** to someone else
- A domain is created by registering it with a **registrar**
  - The registrar tells the top-level domain's operator (the **registry**) which name servers are authoritative for the domain
  - Those servers may be run by the owner, the registrar, or a separate DNS hosting company

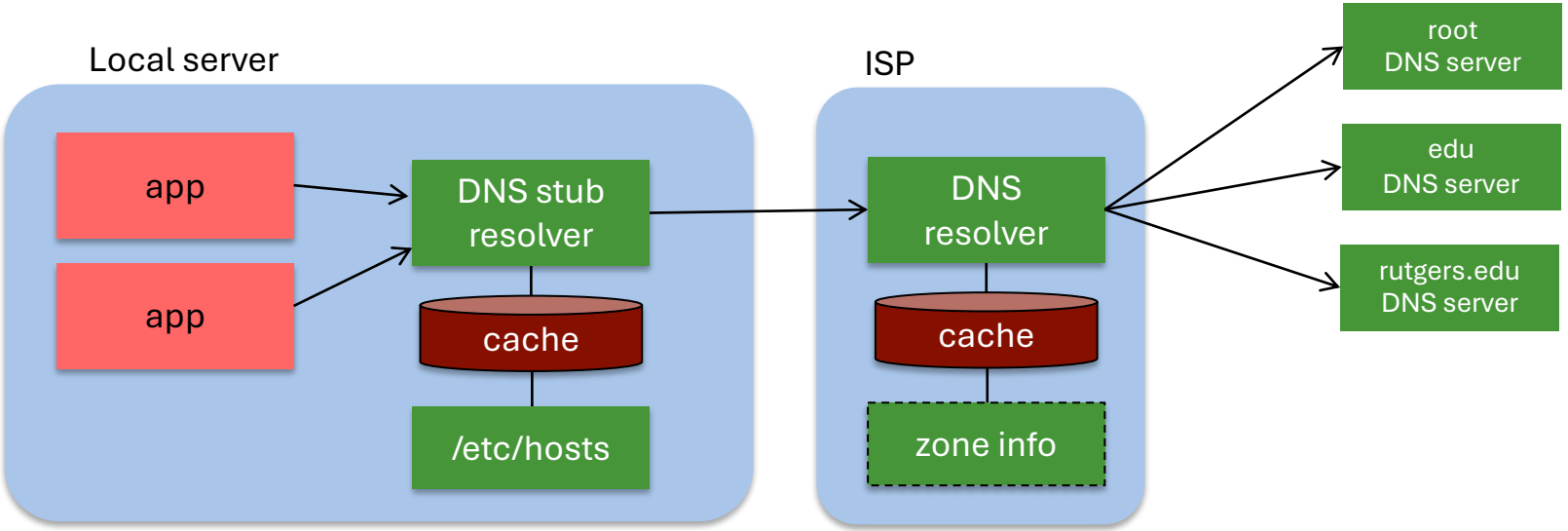
Root Name Servers

- A lookup with nothing cached starts at the root
  - The root servers know which servers handle each top-level domain
- 13 named root server identities, run by 12 organizations
  - Each identity is served by many physical machines around the world



Download the latest list at http://www.internic.net/domain/named.root

How A DNS Lookup Works



Local stub resolver:
- check local cache
- check local hosts file
- send request to external resolver

E.g., on Linux: resolver is configured via the `/etc/resolv.conf` file

External resolver:
- Running at ISP, Cloudflare, Google Public DNS, OpenDNS, etc.

What DNS Trusts

- A traditional query is a **UDP message**:
  - name, record type, and a 16-bit **transaction ID**

- **These fields match replies to queries. They are not authentication**
  - Anyone who can guess or observe them can produce an accepted reply.
  - A UDP source address can be forged
- Queries and replies travel in the clear
- Programs and users trust the result

Pharming

Goal:
Send a user who typed the correct name to a server the attacker runs

- Corrupts name resolution rather than tricking the user into clicking a link
  - The user types the bank's name correctly, or follows a bookmark, and still arrives at the attacker's server
  - A play on phishing: the attacker farms victims instead of fishing for them one at a time
- Done by changing which name server the victim's machine asks, or what that server says

Five Places To Redirect A Lookup

- **The hosts file**:
  Needs file access on the victim's machine. Consulted before any DNS query
- **The resolver setting**:
  Needs code on the victim's machine. Malware points it at the attacker's resolver
  - 2007 to 2011: about four million machines, redirected through paid advertising
- **The home/office router**:
  Needs its administrative password. Changes the DNS server for every device in the LAN
- **The local network**:
  A rogue DHCP server or evil twin hands out the attacker's DNS server
- **The registrar account**:
  Needs the owner's credentials. Every resolver in the world then asks the attacker
  - 2019: Middle East registrars, used to intercept government email

Defending Against Pharming

- The defenses follow the access
  - Endpoint protection for the hosts file and resolver setting
  - The router's password and firmware
  - The switch controls from the local network attacks
- The registrar account: strong authentication and a **registry lock**
  - The registry will not change the domain's name servers without out-of-band confirmation from the owner
  - An ordinary registrar lock only prevents transfers to another registrar
- TLS limits the usefulness of pharming: t
  - The attacker's server doesn't have a certificate for the name the user typed
  - Unless the user clicks past the warning, the site has no HTTPS, or the attacker obtains a certificate

DNS Cache Poisoning (DNS Spoofing)

Goal:
Plant a forged record in a resolver, so every user of that resolver is redirected

- One forged record affects every client of a shared resolver
- An off-path attacker beats the real reply with a forged one:
  1. Cause the resolver to look up the target name
  2. The resolver queries the authoritative server and waits for a matching reply
  3. Flood the resolver with forged replies, each with the server's source address and a different guess at the fields it checks
  4. If a matching forgery arrives first, it is cached. The real reply is discarded

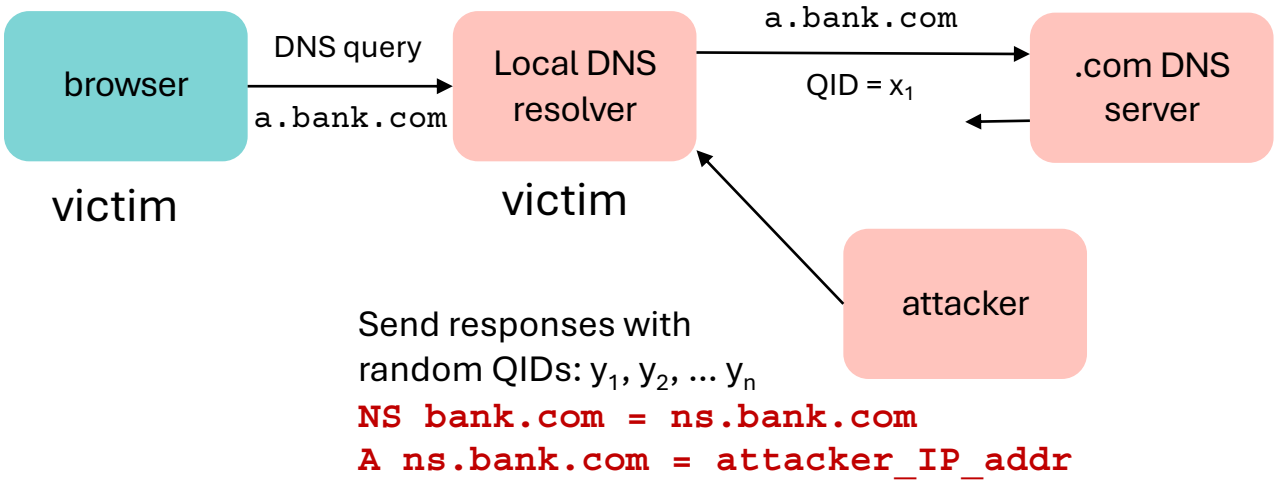
Originally the only unpredictable field was the 16-bit ID, and queries came from a fixed port

The Kaminsky Attack

- Attackers can't poison the DNS cache if a domain is already in the cache
  - The system won't send a DNS query
  - 2008: Dan Kaminsky found a way around the cache
- Query names that do not exist: `aaaa.bank.example`, `aaab.bank.example`, ...
  - Each is a new domain, so the query goes to the authoritative server:
    A new chance for the attacker win
  - The attacker can create responses thousands of times a second with different IDs
- The forged reply is a **referral**: "*`bank.example` is served by `ns1.bank.example`, at this address*" (the attacker's)
  - A referral can contain the zone's name server's address, which gets cached
  - From then on, the resolver asks the attacker for every name in the domain.
  - Getting one forged reply that arrives first is enough for the attacker

The Kaminsky Attack In Pictures

JavaScript on the attacker's web page launches a DNS attacker



If there is some j such that $x_1 = y_j$ then the response will be cached by the victim

All future DNS queries for anything at bank.com will go to attacker_IP_addr

If it doesn't work … try again with b.bank.com, c.bank.com, etc.

Defenses Against Forged Replies

- **Source port randomization** (the 2008 patch)
  - Each query is sent from a random UDP port, and the reply must arrive there
  - The attacker must guess the port and the transaction ID:
    - This increases the complexity from about 65,000 possibilities to several billion
- **DNSSEC**: signed records let the resolver reject a forged record however it arrived
- Randomization defends against a blind attacker. An on-path attacker sees the port and the ID and doesn't need to guess
  - Against that attacker, only a signed reply helps

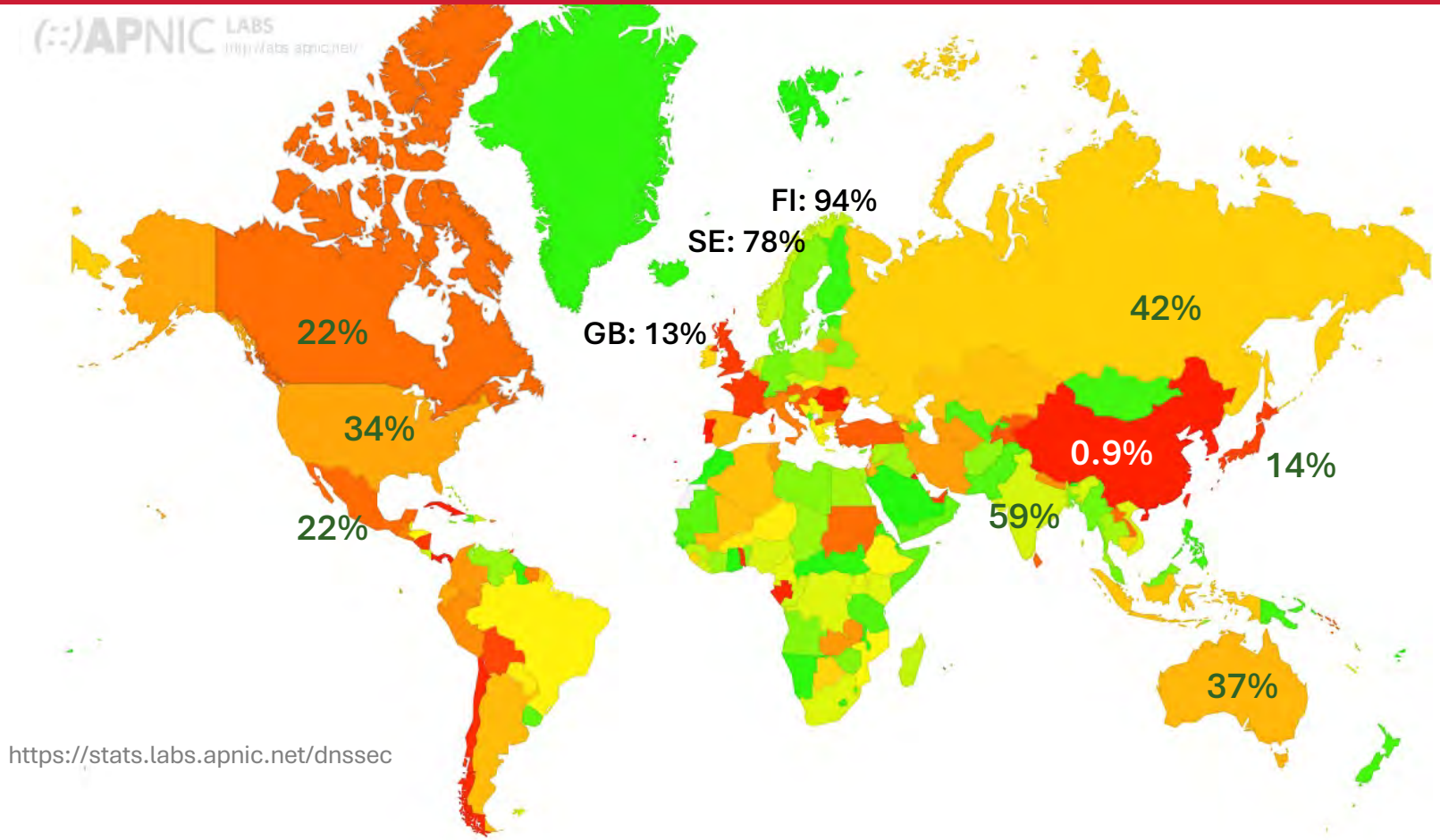
DNSSEC: Signed DNS Records

- **DNSSEC** adds digital signatures to DNS records (standardized 2005)
  - A resolver can verify that a record was published by the zone's owner and not modified
  - It authenticates data. It does not encrypt anything
- The zone owner signs each set of records with the zone's private key and publishes the signatures and the public key
- The parent zone publishes a signed hash of the child's key
  - `edu` vouches for `example.edu`; the root vouches for `edu`
  - A resolver that trusts the root key, the **trust anchor**, verifies the whole chain
  - The root zone was signed in July 2010

What A Validating Resolver Does

- A signed answer whose signatures verify through the chain to the trust anchor is accepted
- An unsigned zone is accepted as it always was
  - Most of the Internet's zones
- A reply from a signed zone whose signatures are missing, expired, or wrong is **bogus** and is discarded
  - The client gets an error rather than a forged address
- That stops cache poisoning for signed zones, and makes DNSSEC mistakes expensive
  - September 2021: Slack's DNSSEC rollout made its service unreachable for hours

DNSSEC Validation By Country, November 2026



What DNSSEC Does Not Cover

- Validation usually happens at the recursive resolver, not on the user's machine
  - The reply from resolver to laptop is an ordinary unsigned message. An attacker on that hop can still forge it
  - Protecting the last hop takes validation on the host or an authenticated connection to a trusted resolver
- DNSSEC authenticates whatever the zone owner published. A malicious domain can sign its own records

Why DNSSEC Adoption Is Low

- **Operational risk**: key rotation must be coordinated with the parent zone
  - An expired signature or mismatched parent record makes the domain bogus for every validating resolver, without a fallback
- **Larger replies**: signatures and keys make replies several times larger
  - Replies that no longer fit in a UDP packet are fragmented, and many firewalls drop fragments
- **Split incentives**: the owner does the work
  - But the benefit reaches only the users whose resolvers validate

DNS Rebinding

Using DNS as designed against the browser's security rule

DNS Rebinding

Goal: let a script on a web page reach devices on the victim's private network

- The browser's **same-origin policy**
  - Designed to keep a script from reading responses from another site
  - Same origin = same scheme, same host name, same port
- The policy compares **names, not addresses**
  - A script from `game.example` may only talk to `game.example`
  - The browser does not care what address that name resolves to right now
  - Addresses change for legitimate reasons, so the browser does not pin them
- If the attacker can change the address:
  - The script can talk to a device inside the victim's network while the browser thinks it is still accessing the same site

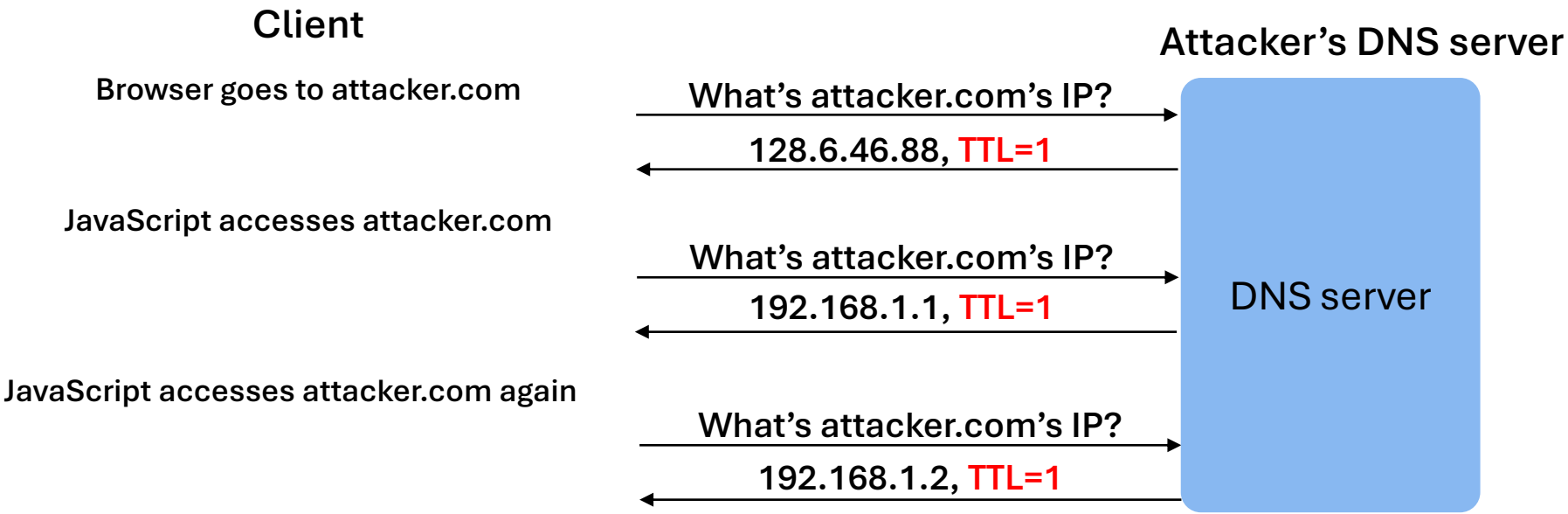
DNS Rebinding: How It Works

The target: an HTTP service inside the network that accepts unauthenticated requests and does not check the host name

1. The victim visits `http://game.example`
   - The attacker's name server answers with the attacker's address and a very short TTL
2. The page loads a script that waits
3. The script requests `http://game.example` again
   - If the browser looks the name up again, it can get `192.168.1.1` (the router) or `127.0.0.1` (the victim's machine)
   - The attacker operates the authoritative DNS server, and controls the response
4. The browser sends the request to the router.
   - Same origin, so the script reads the reply

DNS Rebinding In Pictures

The browser resolves the same name three times and gets a different address the third time, while the same-origin policy sees one site



Defending Against DNS Rebinding

- The strongest defense is on the local service
  - Require authentication for every request
  - Check the `Host` header in the HTTP response
- Resolvers and home routers can refuse to return private or loopback addresses for public names
- Browsers are now treating local network access as a permission
  - Chrome 142 (2025) asks before a public page may send requests to a private address
- Forcing minimum TTLs or pinning addresses helps less:
  - Browsers cache anyway, and legitimate services change addresses

Hijacked Domains: Sitting Ducks Attack

Goal: control an established domain, reputation intact, for phishing and malware

- A domain's NS records point at a DNS hosting provider, but the zone no longer exists there
  - A lapsed subscription, a move to another provider, a mistake years ago: a **lame delegation**
- If the provider lets a new customer claim any domain without verifying ownership, the attacker creates the zone and answers for it
  - The registrar's records are untouched.
- 2024: about 800,000 vulnerable domains, 70,000 hijacked.
- **Defense**: remove stale name servers;
  - Use a provider that verifies control before serving a domain



SECURITYWEEK

CYBERCRIME

Known Brand, Government Domains Hijacked via Sitting Ducks Attacks

Threat actors have hijacked over 70,000 domains, including known brands and government entities, because of failed domain ownership verification

Tens of thousands of domains, including those of well-known brands, non-profits, and government entities, have been hijacked over the past five years because DNS providers failed to properly verify domain ownership, cybersecurity firm Infoblox reports.

The issue was initially disclosed in late July, when Eclypsium and Infoblox said that roughly 35,000 domains had been hijacked since 2018 by abusing the weakness as part of so-called Sitting Ducks attacks.

However, that was just the tip of the iceberg. Infoblox says in a new report. Further investigation into this configuration-oriented attack vector has revealed that at least 800,000 domains could be hijacked, and that 70,000 have already fallen victim to attackers.

"We know these numbers do not accurately reflect the attack surface: they are derived from a limited monitoring system. The challenge with a Sitting Ducks attack is that it is easy to perform and very hard to detect," Infoblox warns in a new report (PDF).

https://www.securityweek.com/known-brand-government-domains-hijacked-via-sitting-ducks-attacks/

Encrypting DNS: DoT And DoH

DNSSEC authenticates records but leaves queries readable to anyone on the path

- **DoT = DNS over TLS** (2016): DNS over a TLS connection on its own port
- **DoH = DNS over HTTPS** (2018): queries as HTTPS requests on the web port
  - Browsers and operating systems support this
- Protects the host-to-resolver hop, the one DNSSEC does not cover
  - The resolver still sees every query
  - A dishonest resolver delivers dishonest replies securely
- DoH has moved much of the world's DNS traffic to just a few companies

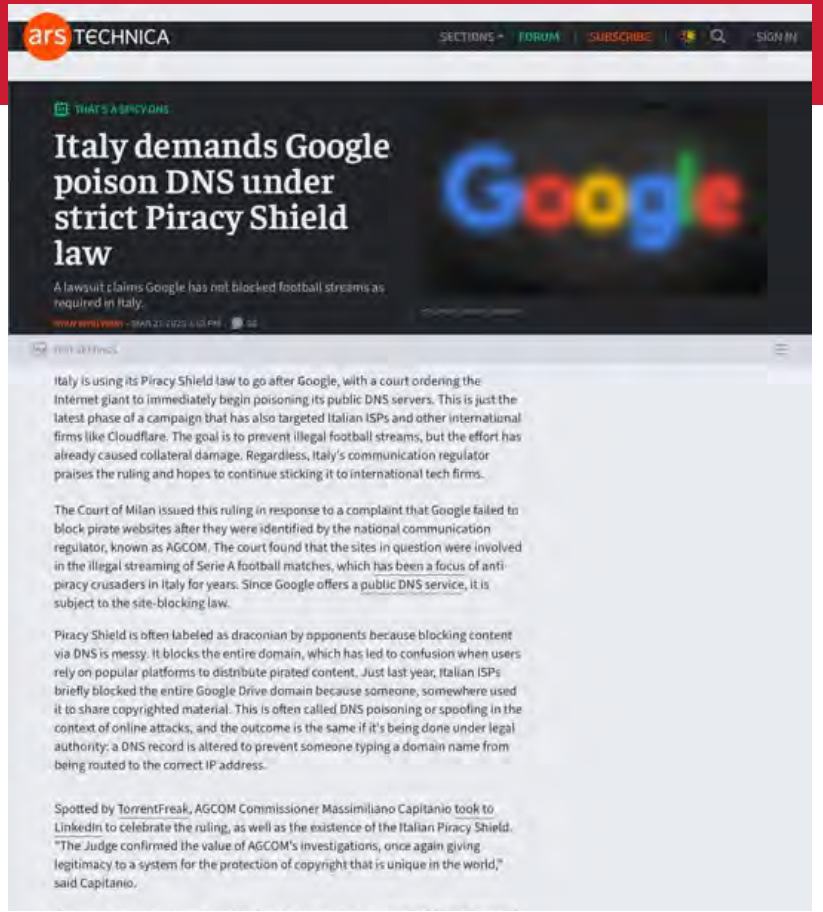
What Each DNS Defense Establishes

Defense	What it establishes	What it does not establish
Random transaction ID and source port	A blind attacker has a much harder guessing problem	Authenticity of the data; protection from anyone who can observe the query
DNSSEC validation	The records were published by the zone's owner and not altered	Confidentiality; that the owner is honest; the resolver-to-host hop
DoT or DoH to a trusted resolver	Privacy and integrity on the host-to-resolver hop	That an unsigned answer is genuine; that the resolver is honest
HTTPS with certificate validation	The server holds the key for a trusted certificate for the requested name	Availability; protection from a fraudulently issued certificate

Deliberate DNS Poisoning

March 2025:

- Italy uses its *Privacy Shield* law to demand Google to poison its DNS servers to prevent people from reaching pirate streams of football games
- A similar case was made against Cloudflare's DNS server and WARP VPN.



ars TECHNICA

SECTIONS FORUM SUBSCRIBE SIGN IN

THAT'S A SPICY ONE

Italy demands Google poison DNS under strict Piracy Shield law

A lawsuit claims Google has not blocked football streams as required in Italy.

Italy is using its Piracy Shield law to go after Google, with a court ordering the Internet giant to immediately begin poisoning its public DNS servers. This is just the latest phase of a campaign that has also targeted Italian ISPs and other international firms like Cloudflare. The goal is to prevent illegal football streams, but the effort has already caused collateral damage. Regardless, Italy's communication regulator praises the ruling and hopes to continue sticking it to international tech firms.

The Court of Milan issued this ruling in response to a complaint that Google failed to block pirate websites after they were identified by the national communication regulator, known as AGCOM. The court found that the sites in question were involved in the illegal streaming of Serie A football matches, which has been a focus of anti-piracy crusaders in Italy for years. Since Google offers a public DNS service, it is subject to the site-blocking law.

Piracy Shield is often labeled as draconian by opponents because blocking content via DNS is messy. It blocks the entire domain, which has led to confusion when users rely on popular platforms to distribute pirated content. Just last year, Italian ISPs briefly blocked the entire Google Drive domain because someone, somewhere used it to share copyrighted material. This is often called DNS poisoning or spoofing in the context of online attacks, and the outcome is the same if it's being done under legal authority: a DNS record is altered to prevent someone typing a domain name from being routed to the correct IP address.

Spotted by TorrentFreak, AGCOM Commissioner Massimiliano Capitanio took to LinkedIn to celebrate the ruling, as well as the existence of the Italian Piracy Shield. "The Judge confirmed the value of AGCOM's investigations, once again giving legitimacy to a system for the protection of copyright that is unique in the world," said Capitanio.

https://arstechnica.com/gadgets/2025/03/italian-court-orders-google-to-block-iptv-pirate-sites-at-dns-level/

The End